IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of: Mark Moir, Victor M. Luchangco, Maurice Herlihy	§	Examiner: Tsai, Sheng Jen
	§	Group Art Unit: 2186
Serial No. 10/620,748	§	
	§	Atty. Dkt. No.: 6000-33800
Filed: July 16, 2003	§	
For: Obstruction-Free Synchronization for Shared Data Structures	§	

REPLY BRIEF

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir/Madam:

This brief is in reply to the Examiner's Answer mailed May 21, 2009. Appellants respectfully request that this Reply Brief be entered pursuant to 37 C.F.R. § 41.41 and considered by the Board of Patent Appeals and Interferences.

REPLY

Appellants respectfully traverse the rejections for at least the reasons given in the Appeal Brief and the following reasons. Different groups of claims are addressed under their respective subheadings. Appellants' arguments from the Appeal Brief filed February 23, 2009 regarding the rejections are herein incorporated by reference.

First Ground of Rejection:

Claims 1-4, 6-10, 12-14, 16-22, 24, 26-33 and 35-44 stand finally rejected under 35 U.S.C. § 102(e) as being anticipated by Martin.

Claims 1, 4, 7-10, 12, 16-18, 24, 26, 29-30, 33, 35, 40 and 43:

1. The cited art clearly fails to anticipate that *the single-target of the single-target synchronization primitive includes a value encoding for an element of the array and a version number encoded integrally therewith.*

In the Appeal Brief, Appellants have argued that none of the passages cited by the Examiner describe a version number, much less a version number that is integrally encoded with a value encoding for an element of the array, or that this encoding is included in the single-target of a single-target synchronization primitive (such as a compare-and-swap - or CAS - operation) used in mediating concurrent execution of access operations, all of which are required by claim 1. In fact, there is no mention of a version number in Martin at all. Instead, Martin teaches the use of CAS with a counter unrelated to a version of an array element (or a version number associated with any other element of Martin), and a distinguishing (i.e., fixed) value RY to which a node's next field may be set using CAS. There is nothing in Martin describing that the V numbers of Martin encode anything other than a value for each array element.

In the Examiner's Answer, p. 17, the Examiner states that he interprets the

limitation "a value encoding for an element of the array" as "a value-encoding mechanism/operation to encode a value for an element of the array," noting that the wording is "value encoding" and not "value encoded." The Examiner submits that a "value encoding" is a mechanism/operation of encoding a value while a "value encoded" is a value that is encoded using the recited mechanism. **Appellants assert that the Examiner's interpretation of the "value encoding" recited in claim 1 is unsupportable in light of the plain language of the claim itself, as well as in light of Appellants' disclosure.** No such mechanism/operation is recited in Appellants' claim, nor would such an interpretation make any sense in the above-referenced limitation. The claim recites "the single-target of the single-target synchronization primitive includes a value encoding for an element of the array…" The target of a synchronization primitive (such as a CAS operation) clearly cannot be or include a mechanism/operation, as suggested, but must be a memory location. Thus, the plain wording of the claim requires that this target (i.e. memory location) includes (i.e. stores) an encoding of a value for an element of the array (the recited "value encoding") and a version number that is integrally encoded with the encoding of the value for the array element. In other words, the plain language of the claim requires that the target of the synchronization primitive includes the two recited elements, "a value encoding" and "a version number" and that the encodings of these two elements are integrated together and included (stored) in the target (memory location). The Examiner's interpretation is clearly not consistent with the plain wording of the claim. Thus, a *prima facie* rejection has not been stated.

In his Answer, the Examiner goes on to state that he interprets the limitation "a version number encoded integrally therewith" as "it is a version number that is the value that is encoded by the value-encoding mechanism/operation recited in the earlier part of the limitation." Appellants again assert that that Examiner's interpretation that the "value encoding" of claim 1 refers to a mechanism/operation is clearly incorrect. Appellants again note that the claim requires that the target include two recited elements, not a single value suggested by the Examiner and taught by Martin.

In the Examiner's Answer, p. 18, the Examiner further submits that Martin

teaches the limitation of "a value encoding process/operation," as values are encoded into each element of the array shown in figure 1 (e.g., V1, V2, V3, etc.) using the encoding process. Appellants again note that Appellants' claim does not recite "a value encoding process/operation" as the Examiner suggests. In fact, the Examiner's remarks are consistent with Appellants' argument that Martin teaches storing a value encoding, but not storing a version number encoded integrally with that value encoding, as required by claim 1.

In the Examiner's Answer, p. 18, the Examiner submits that the claim mentions the "version number" only once, and is completely silent regarding the scope and what constitutes a "version number," (for example, a version number with respect to what attribute) and how the version number is related to the other limitations recited in the claim. Appellants assert that Martin does not teach the integral encoding of any type of version number (with respect to any attribute) encoded together with another element (i.e. a value encoding), as required by Appellants' claim. Appellants further assert that the term "version number" would be easily understood by one of ordinary skill. The concept of a version number is well understood by anyone of ordinary skill in the art and is a distinct requirement from the value encoding. **Martin is completely silent on the subject of versioning.** Appellants again assert that nothing in Martin suggests that the V numbers referenced by the Examiner have anything to do with versioning, and note that the Examiner has not provided any explanation of how the values encoded in Martin could possibly be interpreted as "version numbers". Furthermore, it is clear that the V numbers of Martin cannot reasonably be interpreted as being the two separate elements of Appellants' claim (a value encoding and a version number) integrally encoded together in the target of a single-target synchronization primitive, given the absence of any description whatsoever of any type of version numbers in Martin.

In the Examiner's Answer, p. 19, the Examiner submits that Appellants' FIG. 1 is similar to FIG. 1 of Martin and that it illustrates an invention comprising an array with version numbers denoted as "V1," "V2,"… "Vn." **The Examiner is incorrect.** Appellants' disclosure clearly describes that each element of array 110 includes a value

field 113A (in which the values, V1, V2, etc. are encoded), and a counter field 113B (which indicates a version number). The notation of V1, V2, etc. in Appellants' drawing does not indicate version numbers, and cannot suggest the use of a similar notation to represent version numbers in Martin. **As discussed above, there is absolutely no mention of versioning of the elements of Martin's array or any version numbers at all in Martin.** Instead, the description of FIG. 1 in Martin merely states, "Each node encodes two pointers and a value field" and "Values are represented as "V1", "V2", etc. In general, the value field of the illustrated structure may include either a literal value or a pointer value."

Finally, in the Examiner's Answer, p. 20, the Examiner submits that the word "version" may be defined as "an account from one point of view as opposed to another" and that Martin teaches in paragraph [0012] that V stores different versions of the values of N as a result of CAS operations. This passage describes the syntax of a CAS operation, and states, in its entirety:

> The "compare-and-swap" operation (CAS) typically accepts three values or quantities: a memory address A, a comparison value C, and a new value N. The operation fetches and examines the contents V of memory at address A. If those contents V are equal to C, then N is stored into the memory location at address A, replacing V. Whether or not V matches C, V is returned or saved in a register for later inspection. All this is implemented in a linearizable, if not atomic, fashion. Such an operation may be notated as "CAS(A, C, N)".

The Examiner submits, "Hence, V represents a particular version of N values associated with each CAS operation, i.e., from the point of view of one CAS as opposed to another CAS. Thus, under broadest, reasonable interpretation, the element "a version number" may be considered as "a value" that is associated with each element of the array, representing a particular version of N values associated with each CAS operation. Therefore, Martin teaches the cited limitation." **Appellants again assert that the Examiner's interpretation is unsupportable given the plain language of Appellants' claim.** First, it is unclear what the Examiner means by the statement, "V represents a particular version of N values associated with each CAS operation." The notation "N" in paragraph [0012] of Martin clearly does not refer to one of "N" values associated with a

CAS operation, but to a given "new" value to be stored at the location addressed by A if the comparison operation is successful. Therefore, the phrase, "a particular version of N values" is incomprehensible. **Furthermore, this description of the operation of the CAS operation and the Examiner's remarks are irrelevant to the above-referenced limitation of Appellants' claim.** Claim 1 does not recite that a target of a synchronization primitive (such as a CAS operation) includes "a particular version of a value," but explicitly requires that it includes an encoded version number, as well as a value encoding. Specifically, the claim recites that the target of a single-target synchronization primitive (i.e. a memory location) includes a value encoding for an element of the array **and** a version number encoded integrally therewith. Appellants again assert that there is nothing in Martin describing each of the V numbers of FIG. 1 as encoding or representing anything other than a single pointer value or a single literal value, and assert that there is nothing in Martin describing this value as encoding a version number integrally with the value, as in Appellants' claim. Therefore, Martin cannot be said to anticipate claim 1.

"[U]nless a reference discloses within the four corners of the document not only all of the limitations claimed but also all of the limitations arranged or combined in the same way as recited in the claim, it cannot be said to prove prior invention of the thing claimed and, thus, cannot anticipate under 35 U.S.C. § 102." *Net MoneyIN, Inc. v. VeriSign et al.*, Case No. 07-1565 (Fed. Cir., Oct. 20, 2008). Anticipation requires the presence in a single prior art reference disclosure of each and every element of the claimed invention, arranged as in the claim. M.P.E.P 2131; *Lindemann Maschinenfabrik GmbH v. American Hoist & Derrick Co.*, 221 USPQ 481, 485 (Fed. Cir. 1984). The identical invention must be shown in as complete detail as is contained in the claims. *Richardson v. Suzuki Motor Co.*, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989). For at least the reasons discussed above, Martin clearly cannot be said to anticipate claim 1. A *prima facie* rejection has not been established.

For at least the reasons above, the rejection of claim 1 is unsupported by the cited art and removal thereof is respectfully requested. Independent claims 30, 40, and 43

include limitations similar to those discussed above. Therefore, the arguments presented above apply with equal force to these claims, as well.

Claims 2, 21, and 31:

1. The cited art clearly fails to disclose *wherein the concurrent opposing-end access operations are non-interfering for all but boundary condition states of the array*, as recited in claim 2.

Appellants have argued that the cited portion of Martin (paragraph [0104]) does not describe concurrent opposing-end access operations, much less that such operations are non-interfering for all but boundary condition states, as required by claim 2.

In the Examiner's Answer, p. 20, the Examiner submits that Martin shows concurrent opposing-end access operations in FIG. 1, denoted as "Left Hat" and "Right Hat". **The Examiner is incorrect.** The right hat and left hat of Martin are not access operations at all. They are merely identifiers of left and right sentinel nodes in doubly linked list 102 (see, e.g., paragraph [0077]). Thus, a *prima facie* rejection has not been established.

For at least the reasons above, the rejection of claim 2 is unsupported by the cited art and removal thereof is respectfully requested. Claims 21 and 31 include limitations similar to those discussed above. Therefore, the arguments presented above apply with equal force to these claims, as well.

Claims 3, 20, and 32:

1. The cited art clearly fails to disclose *wherein the non-blocking implementation is obstruction-free, though not wait-free or lock-free*, as recited in claim 3.

Appellants have argued that nothing in Martin teaches obstruction-freedom, as this property is described in Appellants' specification.

In the Examiner's Answer, p. 21, the Examiner notes that Appellants define the term "obstruction-free" this way: "A synchronization technique is obstruction-free if it guarantees progress for any thread that eventually executes in isolation," citing paragraph [1006] of Appellants' Specification. The Examiner submits that Martin teaches, "the left and right ends do not interfere with each other until there is one or fewer items in the queue," and "figures 1-14 of Martin clearly illustrate that the left and right operations are able to make progress advancing along the array until trying to access the same element of the array, which is when interference occurs. Thus, Martin teaches that it guarantees progress for any thread that eventually executes in isolation without interference, hence "obstruction-free"." The Examiner again appears to be equating the labels "left hat" and "right hat" to be operations accessing the elements of Martin's doubly linked list. However, as discussed above, they are not access operations, but are merely identifiers of the right and left sentinel nodes of the doubly linked list. Thus, a *prima facie* rejection has not been established.

For at least the reasons above, the rejection of claim 3 is unsupported by the cited art and removal thereof is respectfully requested. Claims 20 and 32 include limitations similar to those discussed above. Therefore, the arguments presented above apply with equal force to these claims, as well.

Claim 6:

1. The cited art clearly fails to disclose *wherein said mediating comprises attempting to increment the version number included in the single-target of the single-target synchronization primitive*, as recited in claim 6.

Appellants have argued that nothing in Martin that describes the target of a CAS or DCAS operation including a version number encoded integrally with a value, much

less that in mediating, an attempt is made to increment such a version number using a single-target synchronization primitive (e.g., CAS). Appellants further note that DCAS (double compare-and-swap) is clearly not a single-target synchronization primitive, as it operates on two different target locations explicitly identified by operands of the instruction.

In the Examiner's Answer, p. 22, the Examiner refers to his remarks directed to claim 1 regarding the teaching of "a version number." Appellants again assert that Martin teaches no such version number. In addition, the Examiner notes that Martin teaches, "The Hat Trick deque requires only a single DCAS for most pushes and pops." However, Appellants' claim does not recite the use of a single synchronization operation, but a single-target synchronization primitive. DCAS is not a single-target synchronization primitive, since (by definition) it targets two different memory locations on which a synchronization operation is to be performed. The Examiner further submits, "the claims never define the scope and what constitutes a "target", thus allowing the interpretation that a complex-target comprising multiple simpler sub-targets to be treated as a unit of "single" target. For example, a double-word variable may be treated as a single target of operation even though it contains two words." Appellants assert that one of ordinary skill in the art would easily understand the meaning of the term "single-target" as it is applied to the synchronization primitive of Appellants' claims. Appellants agree that a "double-word variable," i.e. a variable that occupies two consecutive word-sized memory locations and is addressable using a single address (e.g., identifying the first of the two words), may be considered a single target of an operation configured to take a double-word as an argument. However, the DCAS operation described in Martin does not target on a double-word, but instead targets two different, independently addressed memory locations. Therefore, it could not reasonably be considered to teach the "single-target synchronization primitive" of Appellants' claims. Furthermore, the Examiner has not provided any explanation of how he believes Martin teaches the use of a DCAS operation to increment a version number included in a single-target of a single-target synchronization primitive, since no such version number is taught by Martin or

incremented by a DCAS operation. Thus, a *prima facie* rejection has not been established.

For at least the reasons above, the rejection of claim 6 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 13:

1. The cited art clearly fails to disclose *wherein the opposing-end accesses include opposing-end, push-type accesses; and wherein the boundary-condition states include a nearly full state,* as recited in claim 13.

Appellants have argued that Martin does not describe a "nearly full" boundary-condition state of an array that is indexable as a circular array, as required by claim 13 (as it depends from claim 11).

Appellants note that the Examiner has not included any additional remarks regarding the rejection of this claim in the Examiner's Answer. Instead, in the Examiner's Answer, pages 22-23, the Examiner repeats the arguments stated in the Final Action mailed October 16, 2008 and refers to his arguments regarding claim 11. Thus, a *prima facie* rejection has not been established.

For at least the reasons above, the rejection of claim 13 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 14:

1. The cited art clearly fails to disclose *wherein distinct left null and right null distinguishing values are employed to identify free elements of the array*, as recited in claim 14.

Appellants have argued that the left hat and right hat of Martin referenced by the Examiner do not teach the left null and right null distinguishing values of Appellants' claim.

In the Examiner's Answer, p. 23, the Examiner notes that paragraph [0074] includes the following, "Each node has a left pointer to its left neighbor and a right pointer to its right neighbor. The doubly-linked chain is terminated at its end nodes by a null in the right pointer of the rightmost node and a null in the left pointer of the leftmost node." Appellants assert that this passage describes a single distinguishing null value that can be present in a pointer field of an end node, not distinct left and right null distinguishing values, as claimed. Thus, a *prima facie* rejection has not been established.

For at least the reasons above, the rejection of claim 14 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 19:

1. The cited art clearly fails to disclose *wherein shared storage usage of the deque implementation is insensitive to a number of access operations that concurrently access the deque*, as recited in claim 19.

Appellants previously asserted that the Examiner failed to address this limitation of claim 19 in his rejection of claim 19, referring only to his rejection of claim 1, which does not include a similar limitation.

In the Examiner's Answer, pages 23-24, the Examiner submits that he has cited the "deque" implementation and operation as an instance of showing why the teaching from Martin reads on the limitations recited in claim 1, and that he has cited the "push" and "pop" operations illustrated in figure 1, which he submits are the corresponding "a number of accessing operations", as another instance of showing why the teaching from Martin reads on the limitations recited in claim 1. The Examiner submits, "Thus, for at

least these two instances, the particular limitation recited in claim 19 is also taught." However, these general descriptions of the deque and the push and pop operations of Martin clearly do not teach the specific limitation of claim 19 recited above. Claim 19 recites that the deque implementation is insensitive to a number of access operations that currently access the deque, not just that it supports or includes some number of access operation types. Thus, a *prima facie* rejection has not been established.

For at least the reasons above, the rejection of claim 19 is unsupported by the cited art and removal thereof is respectfully requested.

Claim 22:

1. The cited art clearly fails to disclose *wherein state of the deque is encoded using an array*, as recited in claim 22.

Appellants previously argued that the Examiner failed to address this limitation of claim 22, referring instead to his remarks regarding claim 6, which does not include such a limitation.

In the Examiner's Answer, pages 24-25, the Examiner submits that as claim 6 depends from claim 1, the claim analysis for claim 1 applies to claim 6. Specifically, the Examiner notes that in the analysis for claim 1, he cites the "deque" implementation and operation as an instance of showing why the teaching from Martin reads on the limitations recited in claim 1, and also cites "the array illustrated in figure 1" as an example of the deque implementation and operations. Appellants note that figure 1 does not illustrate an array, but a doubly-linked list, which is explicitly distinguished from previous array-based deque implementations in Martin in at least paragraph [0078], which states, "However, unlike a typical array-based algorithm, which, on exhaustion of pre-allocated storage, must report a full deque, embodiments in accordance with the present invention allow expansion of the linked-list to include additional nodes. In this way, the cost of splicing a new node into the doubly-linked structure may be amortized

over the set of subsequent push and pop operations that use that node to store deque elements." Thus, it appears that Martin actually teaches away from a deque implemented as an array.

The Examiner also submits that the limitation "version number" recited in claim 6 may be reasonably interpreted as the limitation "state" recited in claim 22, because claim 22 is completely silent on the scope and what constitutes "a state." Appellants assert that one of ordinary skill in the art having benefit of Appellants' disclosure could not interpret the "version number" of claim 6 as the "state of the deque" recited in claim 22. In addition, the plain language of the claims does not support such an interpretation. For example, claim 6 recites a version number encoded together with a value in a target of a single-target synchronization primitive (one of the array elements), and has nothing to do with the state of the array itself. Finally, as discussed above, Martin does not teach the version number of Appellants' claim 6. Thus, a *prima facie* rejection has not been established.

For at least the reasons above, the rejection of claim 22 is unsupported by the cited art and removal thereof is respectfully requested.

Claims 27 and 41:

1. The cited art clearly fails to disclose *wherein the non-blocking deque implementation does not guarantee that at least one of the interfering concurrently executed access operations makes progress*, as recited in claim 27.

Appellants have argued that the Examiner's remarks regarding the left and right hat of Martin do not teach the obstruction-freedom of Appellants' claims, much less that the deque implementation of Martin does not guarantee that at least one of the interfering concurrently executed access operations makes progress.

The Examiner does not include any additional remarks directed to these claims or Appellants' argument in the **Examiner's Answer**, instead referring to his remarks regarding claims 3, 20, and 32. The Examiner's remarks regarding claims 3, 20, and 32 are addressed above. A *prima facie* rejection has not been established.

Claims 28, 37, 38, 39, 42, and 44:

1. The cited art clearly fails to disclose *wherein a separate contention management facility is employed to ensure progress in a concurrent computation that employs the deque implementation*, as recited in claim 28.

Appellants have argued that the DCAS operation described in Martin clearly cannot be a separate contention management facility employed to ensure progress in a concurrent computation that employs the deque implementation, since it is part of the deque implementation itself.

In the Examiner's Answer, pages 25-26, the Examiner notes that the abstract of Martin teaches, "The Hat Trick deque requires only a single DCAS for most pushes and pops. The left and right ends do not interfere with each other until there is one or fewer items in the queue, and then a DCAS adjudicates between competing pops." Appellants again assert that this passage does not teach a separate contention management facility, as claimed. Instead, it refers to a DCAS operation that is part of the deque implementation, as described in at least paragraph [0089], which states, "FIG. 6 illustrates the DCAS operations of competing pop_right and pop_left operations. Because of the semantics of the DCAS only one instance can succeed and the other necessarily fails."

The Examiner further asserts, "First, it is noted that normally when the left and right ends do not interfere with each other the DCAS is not even included in the operation, and that the DCAS is used only when there is one or fewer items in the queue. Thus, DCAS is absent, hence separated, from the operations, most of the time when there is no interference, and only included when there is one or fewer items in the queue.

Second, when there is one or fewer items in the queue the DCAS is used to adjudicates between competing pops, hence a contention management facility." **The Examiner's remarks are completely unsupported in the reference itself. Thus, the Examiner's conclusion is clearly incorrect.** In Martin, all push and pop operations on the deque are implemented using a DCAS operation (see, e.g., paragraphs [0067-0071]. When there is no contention, they succeed. When there is contention between two access operations, one of the DCAS operations succeeds and another fails. This clearly does not teach a separate contention management facility to ensure progress in a concurrent computation that employs the deque implementation, as claimed. Thus, a *prima facie* rejection has not been established.

For at least the reasons above, the rejection of claim 28 is unsupported by the cited art and removal thereof is respectfully requested.

Second Ground of Rejection:

Claims 5, 25 and 34 stand finally rejected under 35 U.S.C. § 103(a) as being unpatentable over Martin in view of Rowlands.

1. The cited art clearly fails to teach or suggest *wherein the single-target synchronization primitive employs a Load-Linked (LL) and Store-Conditional (SC) operation pair*, as recited in claim 5.

Appellants have argued that Rowlands does not teach the specific use of the LL/SC operation pair recited in claim 5.

In the Examiner's Answer, pages 26-27, the Examiner merely asserts the unsupported conclusion that the combination of references teaches the above-referenced limitation of claim 5. However, the Examiner did not explain how this is the case. Appellants can find not teaching in the cited references, whether considered individually

or in combination, that discloses that the single-target synchronization primitive employs a Load-Linked (LL) and Store-Conditional (SC) operation pair, as recited in claim 5.

2. The Examiner has not stated a valid reason to combine the references to teach the limitations of claim 5.

Appellants have argued that no evidence of record describes that the use of LL/SC operation pairs, rather than CAS operations, would enhance the throughput of Martin's system, as suggested by the Examiner. Appellants again assert that the CAS operation described in Martin already allows other processors to access memory locations for read-modify-write operations. Therefore, no other mechanism would be required to allow this access in the system of Martin.

In the Examiner's Answer, p. 27, the Examiner notes that both references are directed toward synchronization mechanism/primitive for multiple accesses competing for the same memory resource. The Examiner submits that he has cited at least one motivation why Rowlands' teaching of LL/SC synchronization mechanism is advantageous, "as it allows other processors to access the memory location for which the atomic read-modify-write is being attempted, which is not permitted in other synchronization mechanisms such as atomic read-modify-write" citing Rowlands, paragraph 0008. The Examiner's remarks are unsupported in the cited reference and are also nonsensical. An LL/SC operation pair **is itself** an atomic read-modify-write mechanism, as described in the paragraph cited by the Examiner, "Another approach for providing an atomic read-modify-write mechanism is the load-linked/store conditional mechanism." Therefore, the Examiner's stated reason to combine the references is not valid. Thus, a *prima facie* rejection has not been established.

For at least the reasons above, the rejection of claim 5 is unsupported by the cited art and removal thereof is respectfully requested.

Third ground of rejection:

The Examiner rejected claims 11, 15 and 23 under 35 U.S.C. § 103(a) as being unpatentable over Martin in view of Latour. Appellants traverse the rejection of these claims for at least the following reasons.

Claims 11, 15, and 23:

1. The cited art clearly fails to teach or suggest *wherein the array is indexable as a circular array*, as recited in claim 11.

Appellants have argued that the cited reference to a circular array of mutexes managed by locks clearly does not teach or suggest the use of a circular array in the double-ended array implementation of Appellants' claims, in which the array implementation is non-blocking, according to the limitations of Appellants' claim.

In the Examiner's Answer, p. 28, the Examiner notes that Appellants' claim 3 recites "wherein the non-blocking implementation is obstruction-free, though not wait-free or lock-free." The Examiner submits, "Thus, it is clear that Appellants' claimed limitation does not require lock-free conditions. Thus, the fact that Latour may have used locks in their invention is totally irrelevant to the limitations associated with claim 11." Appellants assert that the plain language of the claim requires that the implementation be non-blocking. There is nothing in Latour that describes an implementation that is non-blocking, with or without locks. Moreover, Latour's circular array of mutexes are not described as a double-ended array that is indexable as a circular array. Thus, a *prima facie* rejection has not been established.

2. The Examiner has not stated a valid reason to combine the references to teach the limitations of claim 11.

Appellants have argued that there is nothing in the evidence of record that teaches or suggests that the system of Martin would benefit from flexibility in the number of storage locations provided by a circular linked list, or that the deque implementation disclosed in Martin does not include similar flexibility as to the number of storage locations. **Appellants have also argued that it appears that storage locations can be added to the deque implementation of Martin using the "add_right" operation (see, e.g., paragraphs [0116-0121]).** Therefore, there would be no reason to modify Martin to use the circular linked list of Latour, as suggested by the Examiner. Thus, a *prima facie* rejection has not been established.

The Examiner does not provide any additional remarks regarding this argument in the **Examiner's Answer**. Instead, he merely notes that both references are directed toward synchronization mechanism/primitive for multiple accesses competing for the same memory resource, and repeats remarks made in the Final Action mailed October 16, 2008. These remarks have been addressed in Appellants' Appeal Brief.

For at least the reasons above, the rejection of claim 11 is unsupported by the cited art and removal thereof is respectfully requested.

CONCLUSION

For the foregoing reasons, it is submitted that the Examiner's rejection of claims 1-44 was erroneous, and reversal of his decision is respectfully requested.

The Commissioner is authorized to charge the appeal brief fee and any other fees that may be due to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/6000-33800/RCK.

Respectfully submitted,

/Robert C. Kowert/
Robert C. Kowert, Reg. #39,255
Attorney for Appellants

Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
(512) 853-8850

Date: July 21, 2009